API文档

PyJindo 简介

PyJindo 的目的是提供高效的 OSS 访问接口，API 设计对标 oss2。 后文将介绍 PyJindo 目前支持的 OSS API，其中 get_object 性能要显著优于 oss2，单个大文件读取性能可提高数倍。 对于使用 oss2 的用户，如果想要用最少的更改迁移到 PyJindo，只需要安装好 pyjindo 扩展库，然后用：

- import pyjindo as oss2

替代：

- import oss2

PyJindo API 限制

对于已支持的 API，用法与对应 oss2 的 API 类似，但是暂未支持用户指定 HTTP 头部 (headers 参数) 与回调函数 (progress_callback)。 另外，返回值暂未支持 HTTP 请求的相关元数据，如 request_id、versionid 等。

支持的 python 版本

目前支持 python 3.5 至 3.9 版本，其他 python3 版本与 python2 版本待后续支持

基础类

class pyjindo.**Auth**(*access_key_id*, *access_key_secret*)
class pyjindo.**Bucket**

文件（Object）相关操作

上传

Bucket.**put_object**()

上传文件或数据到OSS

用法:

```
>>> bucket.put_object('readme.txt', 'content of readme.txt')
>>> with open('local_file.txt', 'rb') as f:
>>>     bucket.put_object('remote_file.txt', f)
```

Parameters:
- **key** (*str*) – 上传到OSS的文件名
- **data** (*bytes，str 或 file-like object*) – 待上传的内容。

Returns: **PutObjectResult**

Bucket.**put_object_from_file**()

上传一个本地文件到OSS

用法:

```
>>> bucket.put_object_from_file('remote_file.txt', 'local_file.txt')
```

Parameters:
- **key** (*str*) – 上传到OSS的文件名
- **filename** (*str*) – 本地文件名，需要有可读权限

Returns: **PutObjectResult**

下载

Bucket.**get_object**()

下载一个文件

用法:

```
>>> result = bucket.get_object('readme.txt')
>>> print(result.read())
```

Parameters:
- **key** – 文件名
- **byte_range** – 指定下载范围

Returns: file-like object，对返回值执行 read() 将读取数据

Bucket.**get_object_to_file**()

下载一个文件到本地文件

用法:

```
>>> result = bucket.get_object_to_file('readme.txt', 'local_file.txt')
>>> print(result.read())
```

Parameters:
- **key** – 文件名
- **filename** – 本地文件名。要求父目录已经存在，且有写权限。
- **byte_range** – 指定下载范围

Returns: file-like object，返回时文件已写入到本地

Bucket.**append_object**()

追加上传一个文件

Parameters:
- **key** (*str*) – 新的文件名，或已经存在的可追加文件名
- **position** (*int*) – 追加上传一个新的文件， *position* 设为0；追加一个已经存在的可追加文件， *position* 设为文件的当前长度。 *position* 可以从上次追加的结果 *AppendObjectResult.next_position* 中获得。
- **data** (*str 或 bytes*) – 用户数据

Returns: **AppendObjectResult**

拷贝

Bucket.**copy_object**()

拷贝一个文件到当前 Bucket

Parameters:
- **source_bucket_name** – 源文件所在 Bucket
- **source_key** – 源文件名
- **target_key** – 目标文件名

Returns: **PutObjectResult**

删除

Bucket.**delete_object**()

删除一个文件

用法:

```
>>> bucket.delete_object('remote_file.txt')
```

Parameters: **key** (*str*) – 文件名

Returns: **RequestResult**

Bucket.**batch_delete_objects**()

批量删除文件

用法:

```
>>> bucket.batch_delete_objects(['remote_file_1.txt', 'remote_file_2.txt'])
```

Parameters: **key_list** (*list of str*) – 文件名列表，不能为空。
Returns: **BatchDeleteObjectsResult**

重命名

Bucket.**rename**()

Parameters:
- **src** (*str*) – 源路径
- **dst** (*str*) – 目标路径

Returns: **RequestResult**

新建目录

Bucket.**mkdir**()

创建一个目录

Parameters: **key** (*str*) – 目录名
Returns: **RequestResult**

罗列

Bucket.**list_objects**()

根据前缀罗列 Bucket 里的文件

用法:

```
>>> it = ObjectIterator(bucket, prefix="remote_dir")
>>> for info in it:
>>>     print(info.key)
```

Parameters:
- **prefix** (*str*) – 只罗列文件名为该前缀的文件
- **delimiter** (*str*) – 分隔符。可以用来模拟目录
- **marker** (*str*) – 分页标志。首次调用传空串，后续使用返回值的 next_marker
- **max_keys** (*int*) – 最多返回文件的个数，文件和目录的和不能超过该值

Returns: **ListObjectsResult**，可通过 ObjectIterator 创造迭代器

获取文件信息

Bucket.**object_exists**()

文件是否存在

用法:

```
>>> bucket.object_exists('remote_file.txt')
```

Parameters: **key** – 文件名
Returns: True|False

Bucket.**get_object_meta**()

获取文件基本元信息

用法:

```
>>> bucket.get_object_meta('remote_file.txt')
```

Parameters: **key** – 文件名
Returns: **GetObjectMetaResult**，含有`last_modified`,`content_length`,`etag`

返回值

请求信息

class pyjindo.**RequestResult**(*code=None*, *msg=None*, *duration=None*)

Variables:
- **status** – request status. 200 means OK. If not 200, check msg to get more information.
- **code** – error code. 'zero' means no error
- **msg** – error message. 'None' means no error
- **duration** – duration (a string), how long did this request take to return

获取文件元数据

class pyjindo.**GetObjectMetaResult**(*base*, *inode*)

Variables:
- **last_modified** – latest modified time
- **content_length** – file size
- **etag** – etag

class pyjindo.**HeadObjectResult**(*base*, *inode*)

Variables:
- **last_modified** – latest modified time
- **content_length** – file size
- **etag** – etag

下载文件的句柄，有 read 方法

class pyjindo.**GetObjectResult**(*base*, *inode*, *stream*)

追加写文件，含有当前写入位置，可用来继续追加写

class pyjindo.**AppendObjectResult**(*base*, *inode=None*, *next_position=None*)

Variables: **next_position** – position for the next append, which is also the current length of the file

上传文件

class pyjindo.**PutObjectResult**(*base*, *inode=None*)

Variables: **etag** – etag

批量删除文件

class pyjindo.**BatchDeleteObjectsResult**(*base*, *keys=None*)

Variables: **deleted_keys** – list of object keys that were deleted

罗列，通常与 ObjectIterator 配合使用

class pyjindo.**ListObjectsResult**(*base*, *inodes=None*, *objects=None*, *prefixes=None*, *nextMarker=''*, *isTruncated=False*)